Human Interaction and Emerging Technologies (IHIET 2023), Vol. 111, 2023, 224–234
https://doi.org/10.54941/ahfe1004028

Unravelling Scenario-Based Behaviour of a Self-Learning Function With User Interaction

Marco Stang, Marc Schindewolf, and Eric Sax

Institut fuer Technik der Informationsverarbeitung (ITIV), Karlsruhe Institute of Technology (KIT), Germany

ABSTRACT

The lack of explainability in AI models presents a significant challenge that hinders trust and understanding between machines and humans. Explainable AI (xAI) has emerged as a vital field of research and practice, aiming to address this challenge by providing methods and techniques to enhance the interpretability and transparency of AI models. This paper focuses on enhancing the explainability of AI-based systems involving user interaction by employing various xAI methods. The proposed ML workflow, coupled with global and local explanations, offers valuable insights into the decision-making processes of the model. By unravelling the scenario-based behaviour of a self-learning function with user interaction, this paper aims to contribute to the understanding and interpretability of AI-based systems. The insights gained from this research can pave the way for enhanced user trust, improved model performance, and further advancements in the field of explainable AI.

Keywords: Interpretable AI, Explainable AI (xAI), Local explanations, Global explanations

INTRODUCTION AND MOTIVATION

The fields of Artificial Intelligence (AI) and Machine Learning (ML) have gained significant attention and interest due to their remarkable advancements in recent years. With increasing computational power and the ability to learn, reason, and adapt, AI has demonstrated its potential to achieve impressive results across various domains such as banking, automotive, healthcare, and medicine (Goodman and Flaxman, 2017; Stang *et al.*, 2022). Despite the high accuracy achieved by AI models in making predictions, their interpretability is limited due to their capacity to learn from multidimensional data. This has earned them the reputation of being "black boxes" (Wojciech Samek, Thomas Wiegand and Klaus-Robert Müller, 2017). The opacity of AI models raises concerns about understanding the rationale behind their decisions, particularly when these decisions have significant consequences for human lives (Goodman and Flaxman, 2017). For instance, AlphaGo, a deep neural network developed to play the game of Go, has achieved remarkable success by defeating world champions. However, some individual moves by AlphaGo were not interpretable by humans, but still resulted in a game win. Conversely, if a decision leads the model to lose, it may

not have a significant impact on human lives. Nevertheless, the lack of understanding and validation in AI decision-making is a clear disadvantage. In safety-critical applications like self-driving cars, where human lives are at stake, a single incorrect prediction can lead to disastrous consequences. Therefore, there is an urgent need for trustworthy and explainable AI models in such domains. Explainable Artificial Intelligence (xAI) aims to address this interpretability challenge by making the behaviour of AI models more understandable through explanations and demonstrations (Gunning *et al.*, 2019). Understanding how AI models reach specific outcomes is crucial for establishing trust in their decisions. By providing explanations for each prediction, xAI systems bridge the gap between humans and machines, fostering trust and enabling effective collaboration.

FUNDAMENTALS AND STATE OF THE ART

Artificial Intelligence and Machine Learning

Machine Learning (ML) is a subfield of Artificial Intelligence (AI) that focuses on developing computer programs capable of learning and improving from experience without explicit programming. According to Mitchell, (Mitchell, 1997), ML is concerned with constructing computer programs that automatically improve based on experience. The definition of ML is closely intertwined with that of AI, as described by McCarthy (McCarthy, 2007). AI involves creating intelligent machines, including intelligent computer programs, with a focus on understanding human intelligence and problem-solving. While AI encompasses a broader range of aspects, ML specifically concentrates on the learning process, leveraging specific examples and behaviours. Deep Learning (DL) is a subfield of ML that draws inspiration from the structure and function of the brain. The recent widespread adoption of DL can be attributed to the availability of large amounts of data and high-performance GPU hardware. DL enables models to learn hierarchical features by analyzing data at different levels of abstraction, leading to remarkable achievements in areas such as image and speech recognition, natural language processing, and machine translation.

Explainable AI

xAI aims to enhance the understandability of AI system outcomes for human users. Its objective is to enable users to comprehend, gain confidence in, and effectively manage the development of AI systems. This is accomplished by advancing novel machine-learning techniques that generate models with increased explainability. xAI provides users with explanations to understand the system's capabilities, weaknesses, and behaviour in unseen scenarios, empowering them to identify and rectify errors. The development of new ML systems focuses on improving model interpretability through the integration of human-computer interface capabilities. In recent years, various xAI methods have emerged to address the interpretability of ML and DL algorithms. Adadi and Berrada (Adadi and Berrada, 2018) classify these methods into three categories: intrinsic or post-hoc explanations, model-specific or

model-agnostic explanations, and local or global explanations. Intrinsic and post-hoc explanations recognize the impact of the number of parameters on explainability in machine learning models. Intrinsically interpretable algorithms offer a simple way to achieve explainable AI, although there is a trade-off between explainability and accuracy. Explainability techniques can be categorized as model-specific or model-agnostic. Model-specific explanations are tailored to a specific type of model, leveraging its unique characteristics. Model-agnostic techniques, on the other hand, generate explanations based on the input-output pairs of ML models, independent of the model type. Explanations can also be classified as local or global. Global explanations aim to capture the overall logic of the model and trace decision boundaries, providing a comprehensive understanding of its behaviour. However, achieving global explanations for complex models can be challenging. In contrast, local explanations focus on specific instances and provide interpretable explanations for individual predictions, allowing for a more targeted understanding. These categorizations provide a framework for understanding the different types of XAI methods and their approaches to enhancing interpretability in ML and DL algorithms. In the subsequent section, three methods for achieving explainability will be presented and explained, categorized within the established taxonomy.

LIME

The LIME (Local Interpretable Model-agnostic Explanations) method (Ribeiro, Singh and Guestrin, 2016) is a technique used for describing the predictions of machine learning models, particularly in the context of self-learning systems. LIME aims to provide local interpretability by explaining the predictions of a complex model in terms of simpler, interpretable models. The basic idea behind LIME is to generate "perturbed" versions of input instances and observe how the model's predictions change in response to these perturbations. By sampling and perturbing instances around a particular instance of interest, LIME creates a local neighbourhood of interpretable data points. These perturbed instances are then used to train a simpler, interpretable model, such as linear regression or decision trees, which can capture the underlying relationship between the features and the model's predictions. LIME assigns importance weights to the features based on their contribution to the predictions of the interpretable model. These weights explain the model's decision by highlighting the influential features in the local context. By focusing on the local neighbourhood, LIME provides insights into how the model arrived at a specific prediction for a given instance. One of the key advantages of LIME is its model-agnostic nature, meaning that it can be applied to any type of machine-learning model without requiring knowledge of its internal structure. This makes LIME widely applicable across different domains and enables users to gain insights into the decision-making process of black-box models. However, it's important to note that LIME explanations are inherently local and may not capture the global behaviour of the model. The explanations provided by LIME are specific to a particular instance and

may not generalize well to the entire model or dataset. Additionally, the quality of LIME explanations depends on the choice of the interpretable model and the sampling strategy used to generate perturbed instances.

SHAP

The SHAP (SHapley Additive exPlanations) method (Lundberg and Lee, 2017) is based on cooperative game theory, specifically the concept of Shapley values. In cooperative game theory, Shapley values quantify the contribution of each player to the total payoff of a coalition. In the context of the SHAP method, the players are the input features or variables, and the payoff is the model's output or prediction. The SHAP method calculates the Shapley values for each feature by considering all possible combinations of features and evaluating their contribution to the prediction. It considers both the presence and absence of a feature and compares the model's output with and without that feature. By considering all possible combinations, the SHAP method provides a fair allocation of the contribution to each feature. The calculated Shapley values represent the importance or impact of each feature on the model's prediction. Positive values indicate a positive contribution, while negative values indicate a negative contribution. These values can be used to explain the output of the model by attributing the prediction to the relevant features. The SHAP method also allows for visualizing the feature importance using summary plots or individualized explanations for specific predictions. These visualizations help interpret the model's behaviour and understand why it made a particular prediction. Overall, the SHAP method is a valuable tool for explainability in self-learning systems as it provides a systematic and interpretable way to understand the contribution of input features to the model's output, enhancing trust and transparency in the system.

Morris Sensitive Analysis

The Morris Sensitivity method (Morris, 1991) is a global sensitivity analysis technique used to understand the relative importance of input variables or features in a model. It is commonly used in the field of computer experiments and simulation models. Unlike LIME and SHAP, which provide local explanations for individual predictions, the Morris Sensitivity method focuses on understanding the overall impact of input variables on the model's output. The Morris Sensitivity method assesses the sensitivity of the model by perturbing the input variables within a defined range and observing the resulting changes in the output. It measures the effect of each input variable on the model output by calculating a sensitivity index, which represents the average change in the output caused by varying that variable while keeping the others fixed. This sensitivity index quantifies the importance or influence of each input variable on the model's behaviour.

CONCEPT

This paper aims to present an approach for establishing a test system to evaluate a self-learning comfort function. Additionally, it seeks to enhance the explainability and build trust in these systems by integrating explanatory mechanisms. To provide a comprehensive overview of the proposed methodology, a flow diagram (Figure 1) will be utilized. A systematic review of this flow diagram will be conducted, whereby each step will be thoroughly examined and analyzed.

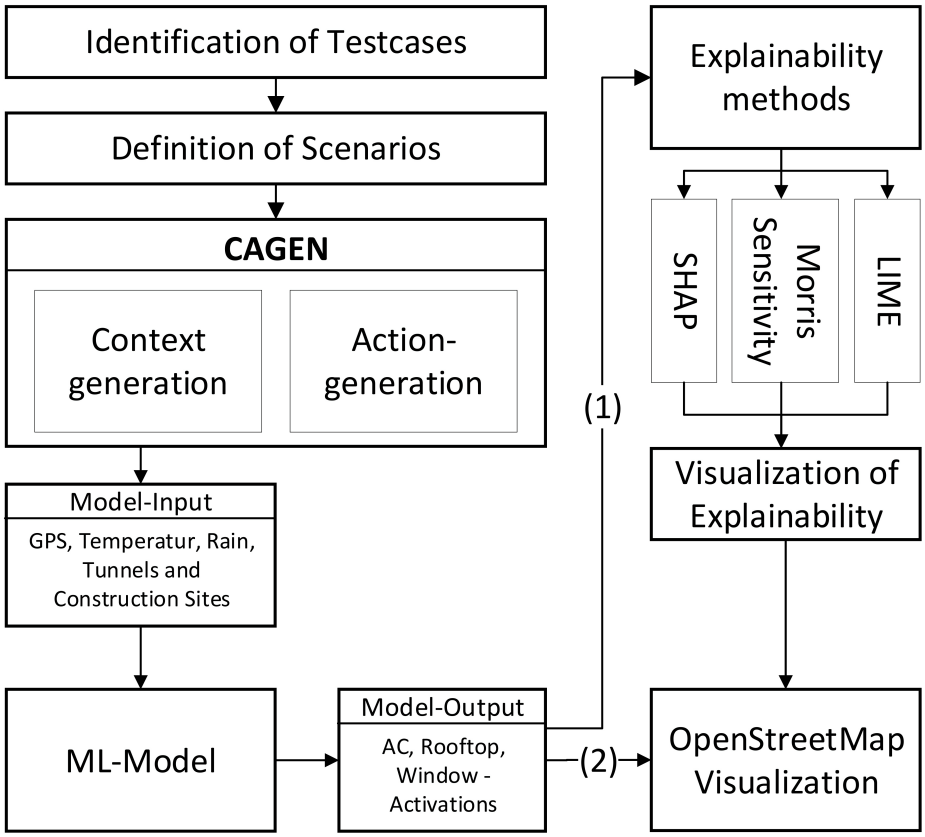

Figure 1: General flow diagram of the proposed concept.

Identification of Testcases introduces test cases as inputs to create scenarios to validate the explainability methods. These test cases can, for example, consist of two similar inputs with only one difference: the presence of rain. In this case, the explainability method is expected to provide an explanation that relates the outcome to the presence of rain. **Scenarios** are established based on the input test cases. An example scenario can be defined as "a working person" who commutes from home to work daily, passing through specific route points such as their child's school, supermarket, and gym. **Context generation with APIs** involves generating context information for the ML model input by utilizing data from the specific scenario. The proposed system examines factors such as the presence of rain, a tunnel, or a construction

site along the route, as well as the temperature throughout the car's trajectory. Real-time APIs are utilized to gather this information, ensuring its validity for testing purposes. **Action generation by user logic** suggests a user-defined logic to determine actions such as controlling the Air Conditioning (AC), rooftop, and windows based on the context information obtained from APIs for the given scenario. The user logic is designed to be straightforward since the primary objective is not to test the user logic itself but to assess whether the model behaves as desired using xAI- methods. The **ML-Model** refers to the ML classifier, which is a Gaussian Process Classifier (GPC). It takes both the context and actions as input and learns from them to classify unseen inputs from different scenarios. The model outputs a file that consists of location points - latitude and longitude - on the related route, context, and actions for every location point. After this point in the flow diagram, the work is branched into two different directions: (**1**) **Explainability branch**, (**2**) **Visualization branch**. The input information for these two layers is the output of the ML-Model, in which the events from previously unseen scenarios are predicted. The **Explainability branch** (**1**) addresses the main motivation for this paper; increasing the explainability of AI-based systems with the help of xAI methods. For this purpose, xAI methods (SHAP, LIME, Morris Sensitivity Analysis) are implemented on previously determined scenarios. Later in this branch, explainability plots are generated for each event that occurs along the route, with the help of stated xAI methods. The **Visualization branch** (**2**) is offering an illustration of the output of the model - including context and actions for every location point - with the help of OpenStreetMap (OSM). The visualization API provided by OSM (OpenStreetMap) offers the capability to visually represent a car driving along a specific route, while effectively demonstrating contextual information and highlighting the corresponding actions on the map.

IMPLEMENTATION OF SCENARIOS AND USER LOGIC

In the initial phase of the research, it is imperative to define the test scenarios that will be used to evaluate the system. These scenarios serve as controlled environments where the performance and behaviour of the system can be assessed. Figure 2 illustrates four distinct test scenarios, each representing a different context. Each scenario within the figure presents a unique combination of tunnel and construction site variations. In this paper, Scenario 4 will be examined in detail.

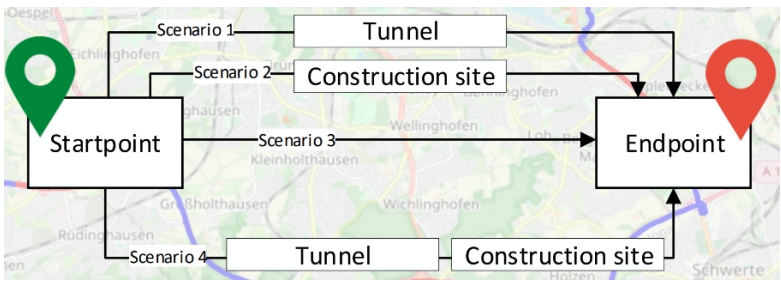

Figure 2: Suggested scenarios with different obstructions.

Scenario 4 has been designed to comprehensively evaluate the capabilities of the ML model, incorporating variations in all features except temperature. This particular scenario presents a sequence of events where the driver encounters distinct environmental conditions and navigational challenges, including passing through a construction site, encountering rainfall, and traversing a tunnel to reach the destination. The data for scenario 4 will be generated using CAGEN. The tool allows the generation of context and action data. A detailed description of the generation of the context data is described by Stang et al., (Stang, Guinea and Sax, 2021) in detail. In the following, the possibility of generating user logic, the interaction of a user with the system, is described. The definition of a user becomes crucial to establish a user logic for training data. The user logic assumes the behaviour of a human and is implemented through nested if-else statements. Common expectations include closing the rooftop in case of rain and cold weather. Similarly, turning on the AC is expected when the temperature is below 10°C. For the window event, a temperature threshold of 12°C is chosen to differentiate it from the rooftop and AC events slightly. The temperature values and the proposed user logic are determined by the author's reasoning. each event (rooftop, window, AC) has its own defined user logic based.

Evaluation of Scenario 4

In this specific scenario, the driver passes by a construction site, encounters rain in the middle of the route, and then enters a tunnel towards the end of the route on a sunny day, where the temperature exceeds 12°C. This test scenario is designed to evaluate the capabilities of the model's features, except for temperature.

Figure 3, left presents the events and corresponding predictions of ML-Model in terms of changes from open to close or close to open actions. The table is structured as follows: The **event** column indicates the number of events that occurred in the order of their occurrence along the route, based on the model's predictions. The **Location** column specifies where each event took place. The **Action** column indicates the action that changed and caused the event to occur. **Previous** and **Next** columns indicate the status change of the corresponding action at the given location. For instance, in event nr. 1 the model predicts that when the construction site is encountered, the status of the window should change from 1 to 0, indicating that the window needs to be closed. By analyzing the table, it can be deduced that events until the tunnel entrance (nr. 13) are correctly predicted by the model when the proposed user logic is considered. The defined user logic can be observed, for example, by looking at event nr. 9: the car encounters rain, and the model predicts that the AC should be set to 1 and turned on. events nr. 13 and nr. 17, however, are predicted incorrectly as the rooftop and window should be closed once the vehicle enters the tunnel, but they remain open. Despite these two events being predicted incorrectly, the subsequent events for both cases are predicted correctly, as the rooftop and windows remain open even when

the car enters the tunnel. Based on this analysis, it is possible to observe the ML model's behaviour following the user logic, as well as identify errors in the ML model. However, it is not possible to explain why the ML model made an error or why it behaved correctly. Therefore, the behaviour will be explained locally for each event point individually, followed by a global explanation.

Event	Position	Action	Previous	Next
1	Start of the construction site	Window	1	0
2	End of the construction site	Window	0	1
3	Start of the construction site	AC	0	1
4	End of the construction site	AC	1	0
5	Start of the construction site	Rooftop	1	0
6	End of the construction site	Rooftop	0	1
7	Start of the rain	Window	1	0
8	End of the rain	Window	0	1
9	Start of the rain	AC	0	1
10	End of the rain	AC	1	0
11	Start of the rain	Rooftop	1	0
12	End of the rain	Rooftop	0	1
13	Entrance of the tunnel	Window	1	1
14	Exit of the tunnel	Window	1	1
15	Entrance of the tunnel	AC	0	1
16	Exit of the tunnel	AC	1	0
17	Entrance of the tunnel	Rooftop	1	1
18	Exit of the tunnel	Rooftop	1	1

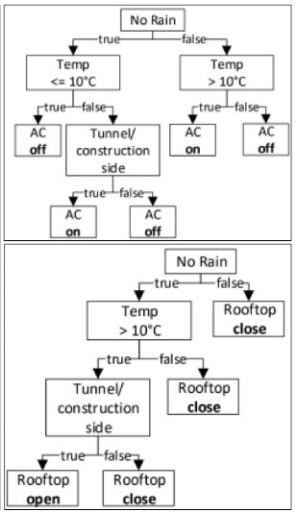

Figure 3: Predictions of the ML model for scenario 4 (left) and proposed user-logic for ac (right, top) and rooftop events (right, bottom).

An explanation of why this event occurred by LIME is given in Figure 4. The prediction probabilities are calculated according to the working principle of the LIME. The figure shows the explanation for event nr. 9. The surrogate model fitted to this specific location point predicts a 0.86% probability for the AC to be turned on. The presence of rain influenced the model to predict that the AC should be turned on, while the absence of a construction site and tunnel still influenced the model to decide that the AC should remain off. Although the model correctly forecasted the event, the low prediction probability could raise concerns regarding trusting in the model's decision. To enhance the model's performance, training data could be augmented by modifying features such as incorporating rain, tunnel, and construction site simultaneously. In Figure 4 (right) the local explanation of event nr. 17 is illustrated. It is detectable from the figure that the predicted output of the model for turning the rooftop on is 58% and the rooftop should be off by 42%. The ML-Modell is unsure of this event and predicts the wrong rooftop state. According to the feature influence, the impact of the feature rain, construction site and temperature is more relevant than the tunnel impact. The same event can also be elucidated using the SHAP method. Figure 5 depicts a so-called force plot generated by the SHAP method for event nr. 17.

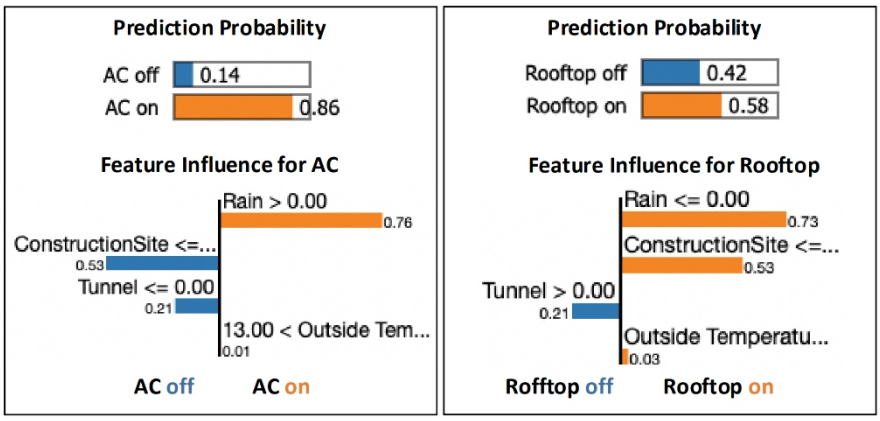

Figure 4: LIME explanations for event nr. 9 (left, correct prediction) and event nr. 17. (left, wrong prediction).

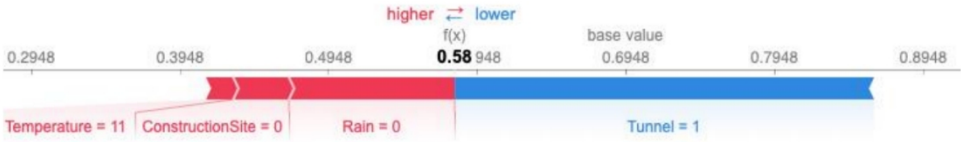

Figure 5: SHAP explanation of event nr. 17.

The base value represents the predicted value that would result if no features were considered. The model's predicted output is 0.58, which corresponds to the predicted value of LIME. The predicted value deviates from the base value (0.6948) due to the influences of the input features. In this event, while the presence of the tunnel lowers the prediction value, correctly indicating the closing of the rooftop, the other features with a value of 0 influence the value to increase. Therefore, the presence of the tunnel alone is insufficient to accurately predict the action, as the combined impact of the other features outweighs the influence of the tunnel alone. For event nr. 17, two distinct were provided. Both explanations lead to the inference that the training data utilized for the model is inadequate for this particular type of test data. A more advantageous training dataset would include a scenario featuring two tunnels: one corresponding to rainy conditions and the other to non-rainy conditions. In this case, rain is chosen as the varying factor since it is the most influential feature contributing to the false prediction. The results of the Morris Sensitivity Analysis provide insights into the average effect of each feature on the model's prediction mechanism. The convergence index of 0.071, representing the analysis's convergence level, indicates the reliability of the results. A lower convergence index suggests better convergence, enhancing the credibility of the analysis. Examining the influence of individual features, the Morris Sensitivity Analysis reveals that rain has the most significant impact on the model's predictions, accounting for approximately 37% of the overall influence. On the other hand, the tunnel feature exhibits the smallest influence among the considered features. This finding contrasts

with the user logic's expectations, as the tunnel feature, which plays a crucial role in the user's decision-making process, demonstrates the lowest degree of influence on the model's output. This observation aligns with the consistently erroneous performance of the model in predicting actions involving a tunnel.

CONCLUSION AND SUMMARY

In conclusion, this paper addressed trust issues between humans and ML models in the automotive industry by focusing on improving interpretability. The objective was to present a workflow encompassing data generation, an ML model for driver behaviour recognition, the introduction of test scenarios, explanation plots using xAI methods, and the evaluation of ML predictions. The application of LIME and SHAP methods for generating local explanations proved to be effective in gaining insights into the model's behaviour for individual events. These methods provided valuable explanations regarding the factors influencing the model's predictions, shedding light on the importance of specific features in determining the outcomes. The analysis conducted in this study highlighted both the strengths and limitations of the ML model in predicting specific events. It emphasized the significance of comprehending the influence of individual features and stressed the need for comprehensive training data that captures the combined impact of multiple factors. Augmenting the training data to include scenarios featuring combinations of influential features, such as rain, tunnel, construction site, and temperature, holds the potential to enhance the model's performance and enable more accurate predictions, particularly in complex situations. By considering these considerations and addressing the identified limitations, such as the underestimated influence of the tunnel feature, it is possible to improve the model's reliability and trustworthiness. This, in turn, can foster greater confidence in ML models within the automotive industry and contribute to safer and more effective decision-making processes.

ACKNOWLEDGMENT

We express our gratitude and appreciation to the student assistant, Umut Ege Uluçay, for his dedicated work and valuable contributions.

REFERENCES

Adadi, A. and Berrada, M. (2018) 'Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)', *IEEE Access*, 6, pp. 52138–52160. doi: 10.1109/ACCESS.2018.2870052.

Goodman, B. and Flaxman, S. (2017) 'European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation"', *AI Magazine*, 38(3), pp. 50–57. doi: 10.1609/aimag.v38i3.2741.

Gunning, D. *et al.* (2019) 'XAI–Explainable artificial intelligence', *Science Robotics*, 4(37), eaay7120.

Lundberg, S. and Lee, S.-I. (2017) *A Unified Approach to Interpreting Model Predictions*. Available at: https://arxiv.org/pdf/1705.07874.

McCarthy, J. (2007) 'What is artificial intelligence'.

Mitchell, T. M. (1997) *Machine Learning*. (McGraw-Hill international editions. Computer science series). New York: McGraw-Hill.

Morris, M. D. (1991) 'Factorial Sampling Plans for Preliminary Computational Experiments', *Technometrics*, 33(2), pp. 161–174. doi: 10.1080/00401706.1991.10484804.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016) *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. Available at: https://arxiv.org/pdf/1602.04938.

Stang, M. *et al.* (2022) 'Development of a self-learning automotive comfort function: an adaptive gesture control with few-shot-learning', *2022 International Conference on Connected Vehicle and Expo (ICCVE)*. IEEE, pp. 1–8.

Stang, M., Guinea, M. and Sax, E. (2021) 'CAGEN - Context-Action Generation for Testing Self-learning Functions', in, pp. 12–19.

Wojciech Samek, Thomas Wiegand and Klaus-Robert Müller (2017) 'Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models'.